Higher Order Non-Homogeneous Helmholtz equation and use of Multigrid V-cycle as a Preconditioner

Student Number: 495026, 767889

Abstract. In this project our team has researched the Helmholtz equation and developed two central difference based 2D stencil finite difference schemes, one with 5 points and one with 9, for a specified BVP (boundary value problem). A FEM scheme is also implemented and applied. Order of converge is shown analytically by Taylor expansion and numerically by converge plots. Numerical optimization strategy and impact of parameter k is discussed.

1. Theory

1.1. About The Helmholtz equation. The Helmholtz equation is a linear elliptic PDE given by

$$\nabla^2 u + k^2 u = -f \tag{1.1}$$

where k is the wavenumber and u is the amplitude of (usually) a physical phenomenon. The homogeneous equation is derived from the wave equation,

$$(\nabla^2 - \frac{1}{c^2}\frac{\partial^2}{\partial t^2})w(x,t) = 0 \ \ , \ \ w(x,t) = u(x)T(t)$$

Using separation of variables. Substituting the above product into the wave equation and simplying, we find

$$\frac{\nabla^2 u}{u} = \frac{1}{c^2 T}\frac{d^2}{dt^2}T = -k^2, \implies \nabla^2 u + k^2 u = 0.$$

which gives the homogeneous Helmholtz equation hence the Helmholtz equation is time-independent, describing a time-harmonic function, i.e. a standing wave, or a collection thereof, on the domain. The PDE can be viewed as an eigenvalue equation. It shows up most often in acoustics, optics, electromagnetism and quantum mechanics.

1.2. Problem. The problem under consideration for the finite difference schemes and FEM (finite element method) is,

$$\nabla^2 u + k^2 u = -f \ \text{ on } \ \Omega = (0,1)^2$$
$$u(0,0) = u(0,1) = u(1,0) = u(1,1) = 0$$

The function f is zero in the homogeneous case, and $f = 20\pi^2 \sin(n\pi x)\sin(m\pi y)$. In quantum dynamics, this BVP may describe the orbitals of the energy state of an electron.

1.3. 5-Point Stencil. We have opted for implementing a 5 point scheme and a 9 point scheme. Given a non-homogenous Helmholtz equation (1.1) on a square domain with homogeneous Dirichlet boundary conditions, using a five point stencil the PDE can be discretized as following,

$$k^2 u_{i,j} + \frac{1}{h^2}\big(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}\big) = -f_{i,j},$$
$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} + (h^2k^2 - 4)u_{i,j} = -h^2 f_{i.j},$$

The above scheme can be written in a form of a linear system as,

$$(h^2k^2\mathbb{I} + L)u = F, \implies Au = F,$$

where $A = h^2k^2\mathbb{I} + L$, $\mathbb{I}$ is the identity matrix, and L is the Laplacian operator.

Since the Helmholtz equation is time harmonic, it should have no issues with stability. As for convergence, we will only show analytically for 9-point stencil.

1.4. 9-Point Stencil. We have implemented a 9-point stencil scheme that uses all gridpoints in a 3x3 region centered at the gridpoint $u_{i,j}$ to be approximated. The 9-point stencil ∇_9^2 for the Laplacian is approximated by

$$\nabla_9^2 = \frac{1}{6h^2}(-20u_{ij} + u^+_{i\pm1,j\pm1} + 4(u^+_{i\pm1,j} + u^+_{i,j\pm1}), \tag{1.2}$$

with $u^+_{i\pm1,j\pm1} = u_{i+1,j+1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i-1,j-1}$. Horizontal term $u_{i\pm1,j}$ has expansion

$$u(x \pm h, y) \approx u(x,y) \pm hu_x(x,y) + \frac{h^2}{2!}u_{xx}(x,y) \pm \frac{h^3}{3!}u_{xxx}(x,y) + ...$$

such that terms will cancel in the horizontal sum

$$4(u_{i+1,j} + u_{i-1,j}) = 8u + 8\frac{h^2}{2!}u_{xx} + 8\frac{h^4}{4!}u_x^{(4)} + 8\frac{h^6}{6!}u_x^{(6)}$$

where is $u_x^{(4)} = u_{xxxx}$. The same reasoning goes for the vertical terms, except with the derivative taken in y-direction. For the diagonal terms, one must differentiate in both directions. Add these up, a lot of terms will cancel, such that

$$\nabla_9^2 = \frac{1}{6h^2}\Big(12\frac{h^2}{2!}(u_{xx} + u_{yy}) + 12\frac{h^2}{4!}(u_x^{(4)} + u_y^{(4)}) + 12\frac{h^2}{6!}(u_x^{(6)} + u_y^{(6)})\Big)$$

$$\nabla_9^2 = \nabla^2 u + \frac{h^2}{12}(\nabla^2(\nabla^2 u)) + \frac{12}{6!}h^4(\partial_x^{(4)}, \partial_y^{(4)}), \tag{1.3}$$

Using the 9-point stencil, we can discretize the non-homogeneous Helmholtz equation as,

$$\nabla_9^2 u = (\nabla^2 u + \frac{h^2}{12}\nabla^2(\nabla^2 u)) + O(h^4) \tag{1.4}$$

Substituting every $\nabla^2 u$ with $-f - k^2 u$ and separating u-terms from f-terms, we get the following.

$$(\nabla_9^2 + k^2 - \frac{k^4 h^2}{12})u = (\frac{k^2 h^2}{12} - 1)f - \frac{h^2}{12}\nabla^2 f \tag{1.5}$$

Now we show that our discretization has fourth order convergence. Substituting ∇_9^2 as $\nabla^2 + \frac{h^2}{12}(\nabla^2(\nabla^2 u))$ as by Equation 1.4, and using Equation 1.3 in Equation 1.5, we find the truncation error as,

$$\begin{aligned}\tau_m &= \Big(\nabla_9^2 + k^2 - \frac{k^4 h^2}{12}\Big)u - \Big(\frac{k^2 h^2}{12} - 1\Big)f + \frac{h^2}{12}\nabla^2 f \\ &= \Big[\nabla^2 u + \frac{h^2}{12}\Big(\nabla^2(\nabla^2 u)\Big)\Big] + \frac{12}{6!}h^4(\partial_x^{(4)}, \partial_y^{(4)}) \\ &\quad + \Big[k^2 - \frac{k^4 h^2}{12}\Big]u - \Big(\frac{k^2 h^2}{12} - 1\Big)f - \frac{h^2}{12}\nabla^2 f\end{aligned}$$

Substituting again $\nabla^2 u$ by $-f - k^2 u$ every time it shows up,

$$\begin{aligned}\tau_m &= -f - k^2 u + \frac{h^2}{12}(-\nabla^2 f - \nabla^2(k^2 u)) + k^2 u - \frac{h^2 k^4}{12}u \\ &\quad + f - f\frac{h^2 k^2}{12} + \frac{h^2}{12}\nabla^2 f + \frac{12}{6!}h^4(\partial_x^{(4)}, \partial_y^{(4)}) \\ &= \frac{12}{6!}h^4(\partial_x^{(4)}, \partial_y^{(4)}) \leqslant \frac{12}{6!}h^4|K| \;\text{ with }\; |K| = max\{\partial_x^{(4)}, \partial_y^{(4)}\}\end{aligned}$$

hence truncation error τ_m is fourth order for the 9-point scheme.

1.5. Weak Formulation for the Finite Element Scheme. To obtain the weak formulation of (1.1) we test the equation against any smooth function $v \in V$ where $V = \{v : v \in C^\infty(\Omega), v(\mathbf{x}) = 0 \text{ for } \mathbf{x} \in \partial\Omega\}$. We multiply our problem with the test function $v \in V$ and integrate,

$$-k^2 \int_\Omega u(\mathbf{x})\, v(\mathbf{x}) d\Omega \; - \int_\Omega \Delta u(\mathbf{x})\, v(\mathbf{x}) d\Omega \; = \int_\Omega f(\mathbf{x})\, v(\mathbf{x}) d\Omega.$$

Using Greens formula on the second term in the above expression, we get,

$$-k^2 \int_\Omega u(\mathbf{x})\, v(\mathbf{x}) d\Omega - \left(\int_{\partial\Omega} \Big(\nabla u(\mathbf{x}).\, \mathbf{n}\Big) \underbrace{v(\mathbf{x})}_{=0} ds - \int_\Omega \nabla u(\mathbf{x})\, \nabla v(\mathbf{x}) d\Omega \right) \; = \int_\Omega f(\mathbf{x})\, v(\mathbf{x}) d\Omega.$$

Using this in the above formulation we are left with the following week formulation. Find $u \in V$ such that for all $v \in V, \Omega = (0,1)^2,\ \mathbf{x} = (x, y)$

$$-k^2 \int_\Omega u(\mathbf{x})\, v(\mathbf{x}) d\Omega \; + \int_\Omega \nabla u(\mathbf{x})\, \nabla v(\mathbf{x}) d\Omega \; = \int_\Omega f(\mathbf{x})\, v(\mathbf{x}) d\Omega, \quad . \tag{1.6}$$

2. Experimental Setup

For the numerical implementation of the Helmholtz equation, we've used MATLAB as the programming language.

2.1. Finite Difference Implementation. Helmholtz equation discretized using a 5-point stencil and the 9-point stencil is implemented using the conjugate gradient method. The tolerance set for the iterative method is 10^{-12} and wave number $k = \sqrt{8/h^2} + 1$ for most of the experiments. An important goal for the implementation had been to optimize the code for better performance, therefore we avoid setting up the matrix A by working directly on the grid. The algorithm for the conjugate gradient can be found in Appendix. Conjugate Gradient method is used when the matrix A is symmetric positive definite and we show positive definiteness by showing the matrix A is strictly diagonally dominant for $k > \sqrt{8/h^2}$, which implies all eigen values of the matrix will be positive and hence the matrix is positive definite. For most of the numerical calculations we used wave number $k = \sqrt{8/h^2} + 1$, however a different choice of wave number gave different results. We also tested our implementations for small value of k such as $k = 1/\pi$, which will be shown and discussed in later section.

2.2. Finite Element Implementation. For a square domain, we obtain the mesh shown in Figure 6 (Appendix) by triangulating in equal triangles. In the numerical implementation, *tri* stores row wise, the data of each triangle's vertices respective to their triangle and P is the geometry matrix that contains the data of points at vertices. As discussed in the theoretical section, we need to setup the stiffness matrix S, the mass matrix M and the load vector F.

2.2.1. *Stiffness Matrix.* The stiffness matrix as formulated in the previous section is given as $S = \sum_K \int_{K_k} \nabla\varphi_i \nabla\varphi_j \, d\Omega$ For all $K \in \Omega$, the stiffness matrix was found by transporting it to reference element and solving it on the reference element. Using chain rule and substitution $x = x(\hat{x})$ and $\varphi = \hat{\varphi}(\hat{x})$ we get,

$$\nabla\varphi_i \nabla\varphi_j \; = \sum_K \frac{\partial\varphi_i}{\partial x_k} \frac{\partial\varphi_j}{\partial x_k} = (\nabla_{\hat{x}}\hat{\varphi}\, J^{-1})\,(\nabla_{\hat{x}}\hat{\varphi}\, J^{-1}) = G\, G^T,$$

From this we observe for linear basis, $\nabla\varphi_i \nabla\varphi_j \; = G\, G^T$ is a constant on K_k, so using the area of the triangle $det(J)/2!$ we can write an explicit expression for the stiffness Matrix.

$$S = \frac{G\, G^T\, |J|}{2}$$

2.2.2. *Mass Matrix.* The mass matrix was solved using Gaussian Quadrature order 4 locally on each triangle and then transported back to their respective position on the global triangle.

2.2.3. *Load Vector.* Load vector was also solved using Gaussian Quadrature order 4. Load vector as formulated in the previous section is written as,

$$F_j \; = \int_{K_k} f\, \varphi_j,$$

This was achieved locally on each triangle using,

$$F_j \; = \frac{|J|}{2} \omega_i \, \lambda_j^i \, f(x(\lambda^i)), \quad j = 1, 2, 3$$

where $\lambda^i = (\lambda_1^i, \lambda_2^i, \lambda_3^i)$ are quadrature nodes in barycentric coordinates and ω are respective quadrature weights. After evaluating load vector locally it was transported to it's respective global position in the mesh.

2.2.4. *Boundary Conditions.* Homogeneous Dirichlet Boundary conditions were implemented by first finding the vertices that are on the boundary say i and then setting $A_{ii} = 1$ and $A_{ij} = 0$ for $i \neq j$. Likewise the load vector was set zero at the boundary vertices i.e $F_i = 0$. After implementing the boundary conditions, the solution was found using backslash command.

3. Results and Discussion

3.1. Convergence. Figure 1a shows the convergence plots, i.e step-size vs error plots for second order and fourth order schemes. The order for calculated for second order scheme and fourth order scheme are 2.05 and 4.04, respectively which are close to the theoretical accuracy. The reference solution for the non-homogeneous problem was calculated using a small step-size of $N = 2048$ grid points by a fourth order numerical scheme.

The next Figure 1b shows a semi-logarithmic plot of the Error vs the iterations plot, needed by conjugate gradient method to convergence to an accuracy of *tol* $= 10^{-12}$. This tolerance number is important and how it affects the solution will be discussed later in this section. It is observed that conjugate gradient method implemented using a fourth order scheme converged to the desired tolerance in *3 iterations*, whereas the second order scheme converged in *261 iterations*. This fact explains, fourth order scheme with conjugate gradient method outperforms the second order scheme remarkably. Figure 1c shows logarithmic plot of time taken by CPU for computation vs the step-size. The results shown in Figure 1b and 1c are relate able. As discussed before it took less iterations for a higher order scheme to converge to a desired tolerance therefore it makes sense that it takes less time for a higher order scheme as well, however the time for both schemes increases by increasing the number of grid points.

3.2. Effect of Wave number k. The choice of wave number was an important for both finite difference schemes and finite element schemes. As discussed by the author [**1**], usual Finite Element schemes gave unstable results for large values of k. In order for the finite element schemes to give satisfactory results, $N = k^d$ number of grid points are required, where k is the wave number and d is the dimension of the problem. Similar results holds for finite difference schemes. We numerically show, how a second order scheme finite difference scheme and finite element method gives unsatisfactory results in the upcoming sections. On the contrary, a small value of k gave accurate results. This can be seen in Figure 5. As seen in the figure, the finite difference and FEM schemes came up with different solutions; FEM made the two furthest wave tops the highest, while finite difference made the closest one higher. This may be caused by a choice of k that makes the solution unable to divide the domain into equally spaced nodes. 'Nodes' here signifies the points on the domain where the solution is zero.

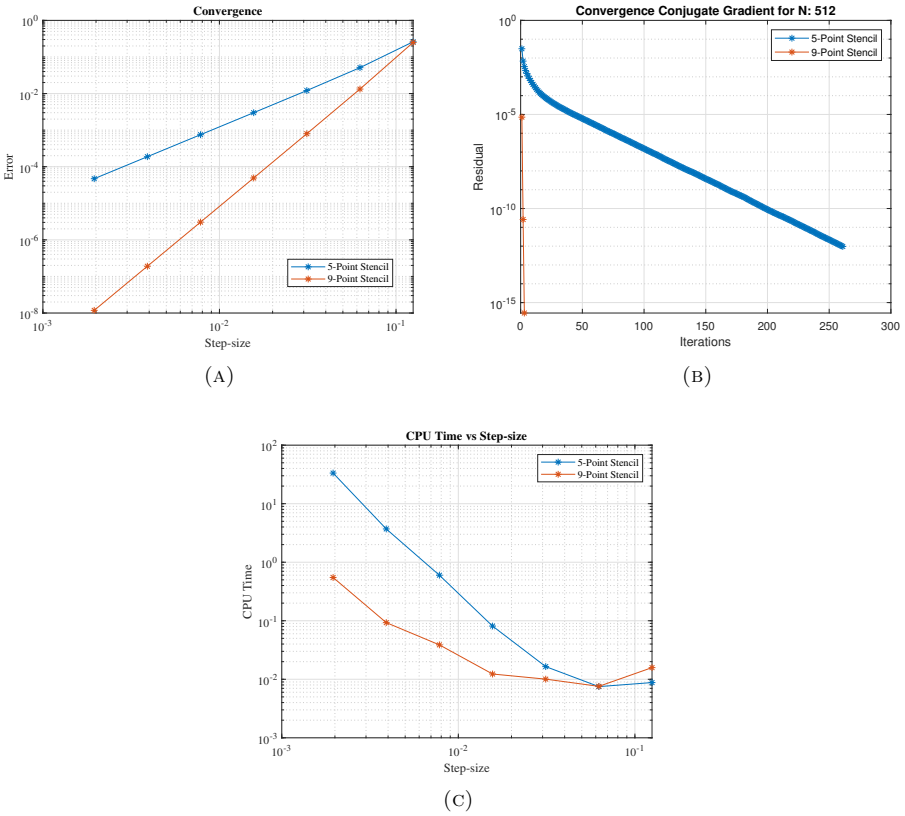
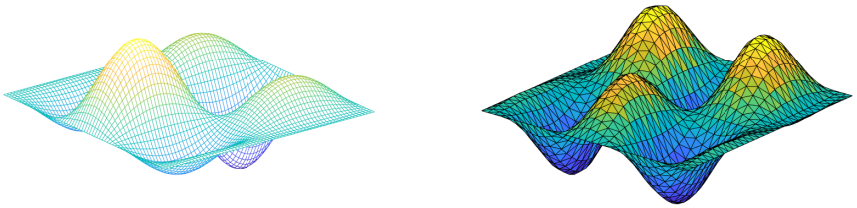


FIGURE 1. *(Figure 1a)* Convergence Plots. *(Figure 1b)* Speed of Convergence for Conjugate Gradient Method. *(Figure 1c)* CPU time for varying Step-sizes

3.3. Instability for large k. Choice of k and tolerance for the CG method was important aspect of calculations as it lead to instabilities in the solution. Figure 3 shows a comparison of fourth order scheme(bottom row) and second order scheme for large tolerance. It is observed that a second order scheme becomes less smooth as the number of grid points is increased, however if the tolerance is set considerably small, i.e 10^{-12}, the second order scheme gives smooth solutions. Similar results were observed for the finite element implementation which can be seen in Figure 4. Our intuition is the wave number k affects the solution by introducing high frequencies when the number of grid points are small causing non-smooth solution. This can be dealt by the using very small grid size, which will make the problem uneconomical in terms of memory and speed and different preconditioners should be then used to speed up the convergence.



(A) (B)

FIGURE 2. *(Left)* Finite Difference Implementation. *(Right)* Finite Element Implementation. Wave number $k = 1/\pi$

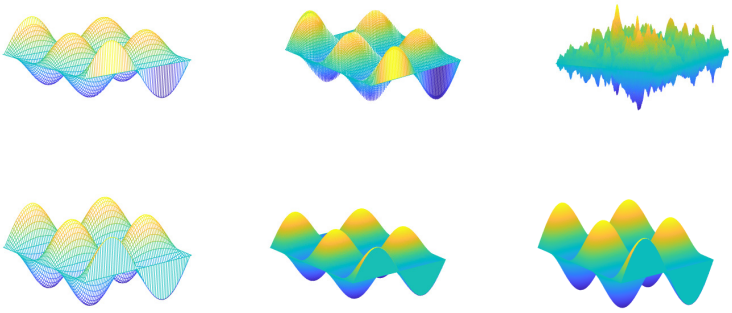


FIGURE 3. Parameters: *tolerance*= 10^{-6}, $k = \sqrt{8/h^2} + 1$, and $N = \{64, 128, 512\}$. *(Top Row)* Second order scheme. *(Bottom Row)* Fourth order scheme.

3.4. Multigrid as preconditioner. In Figure 1b it is observed, discretizations using the 5-point stencil took 261 iterations to converge in contrast to the discretizations using 9-point stencil, which took only 4 iterations. The reason is obvious as 9-point stencil is fourth order accurate it'll work better than 5-point stencil. However, the draw backs with 9-point stencil are it's complexity and difficulty to implement. For a homogeneous problem, 9-point stencil gives fourth order however, in case of a non-homogenous problem there are error dominant force terms that should be taken into account when discretizing the problem else the method

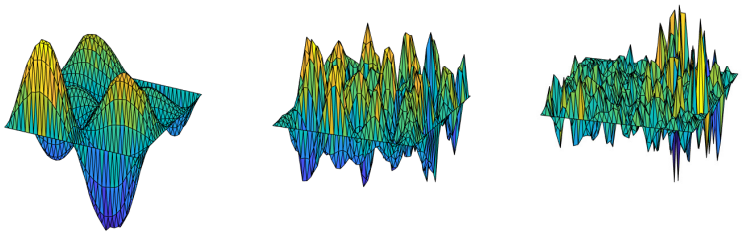
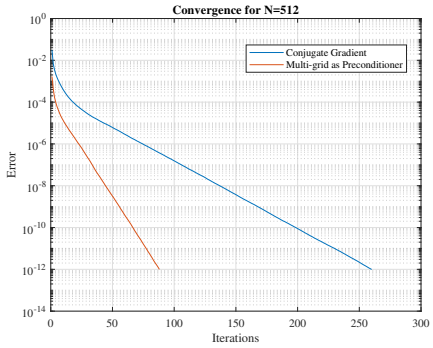


FIGURE 4. Parameters: $k = \{10, 20, 40\}$, and $N = 32$. Instability with increasing wave number. Solution using Finite Element

will be second order accurate. To avoid the complexity and for faster convergence, we implemented multigrid techniques as preconditioner.



(A)

FIGURE 5. Convergence comparison for Conjugate Gradient and Multigrid method used as preconditioner

4. Conclusion

We have solved Helmholtz equation using 2nd and 4th order accurate finite difference, and FEM for linear basis. Solution was highly affected by choice of k and the stopping criteria for the CG method also showed an important role for the second order scheme. We observed a high value of k results in instabilities in the solution which can be overcome by either setting a lower tolerance for the conjugate gradient method or using a higher order scheme such as fourth order 9 point stencil(in case of finite differences). We also implemented different preconditioners and multigrid tests for this problem but due to page restriction we were unable to write them in the report. The work load was distributed as, Adrian did the theoretical work and Muhammad Hamza did the numerical implementations.

References

[1] Esterhazy, S., and Melenk, J. M. On stability of discretizations of the helmholtz equation. In *Numerical analysis of multiscale problems*. Springer, 2012, pp. 285–324.

Appendix A.
Algorithms Conjugate Gradient Method

Compute $r_0 := F - Ax_0$, $p_0 := r_0$,;
for $j = 0, 1, ..$ *Until Convergence* **do**
  $\alpha_j := (r_j, r_j)/(Ap_j, p_j)$;
  $x_{j+1} := x_j + \alpha_j p_j$;
  $r_{j+1} := r_j - \alpha_j A p_j$;
  $\beta_j := (r_{j+1}, r_{j+1})/(r_j, r_j)$;
  $p_{j+1} := r_{j+1} - \beta_j p_j$;
end

Algorithm 1: Conjugate Gradient Method

Appendix B.
Algorithm Preconditioned Conjugate Gradient

Compute $r_0 := F - Ax_0$, $z_0 := M^{-1} r_0$ and $p_0 := z_0$,;
for $j = 0, 1, ..$ *Until Convergence* **do**
  $\alpha_j := (r_j, z_j)/(Ap_j, p_j)$;
  $x_{j+1} := x_j + \alpha_j p_j$;
  $r_{j+1} := r_j - \alpha_j A p_j$;
  $z_{j+1} := M^{-1} r_{j+1}$;
  $\beta_j := (r_{j+1}, z_{j+1})/(r_j, z_j)$;
  $p_{j+1} := z_{j+1} + \beta_j p_j$;
end

Algorithm 2: Preconditioned Conjugate Gradient

Appendix C. Triangulation

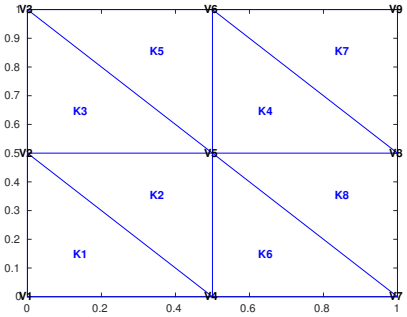


FIGURE 6. Triangulation of square mesh